YALMIP工具箱简介

东北大学数学系
王琪
wangqimath@126.com

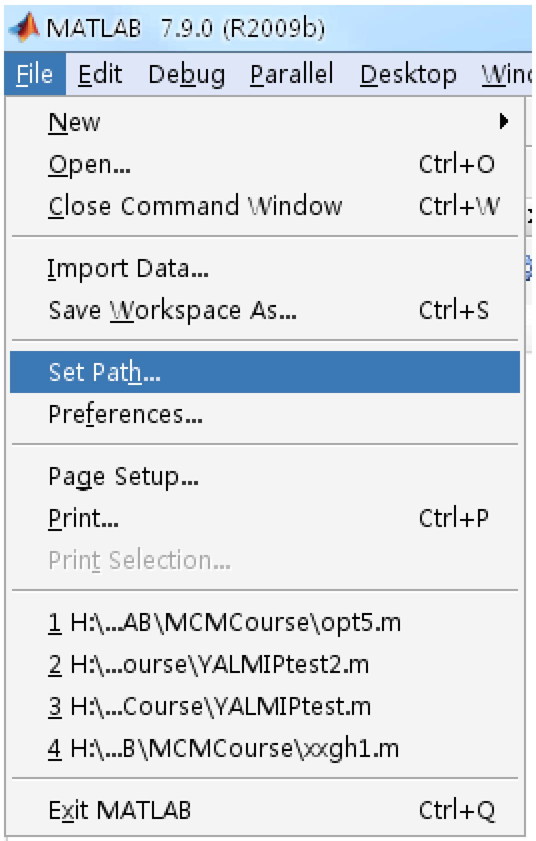
YALMIP工具箱简介

- 基于符号运算工具箱编写
- 一种定义和求解高级优化问题的模化语言
- 用于求解线性规划、整数规划、非线性规划、混合规划等标准优化问题以及LMI
- 工具箱主页：http://control.ee.ethz.ch/~joloef/wiki/pmwiki.php

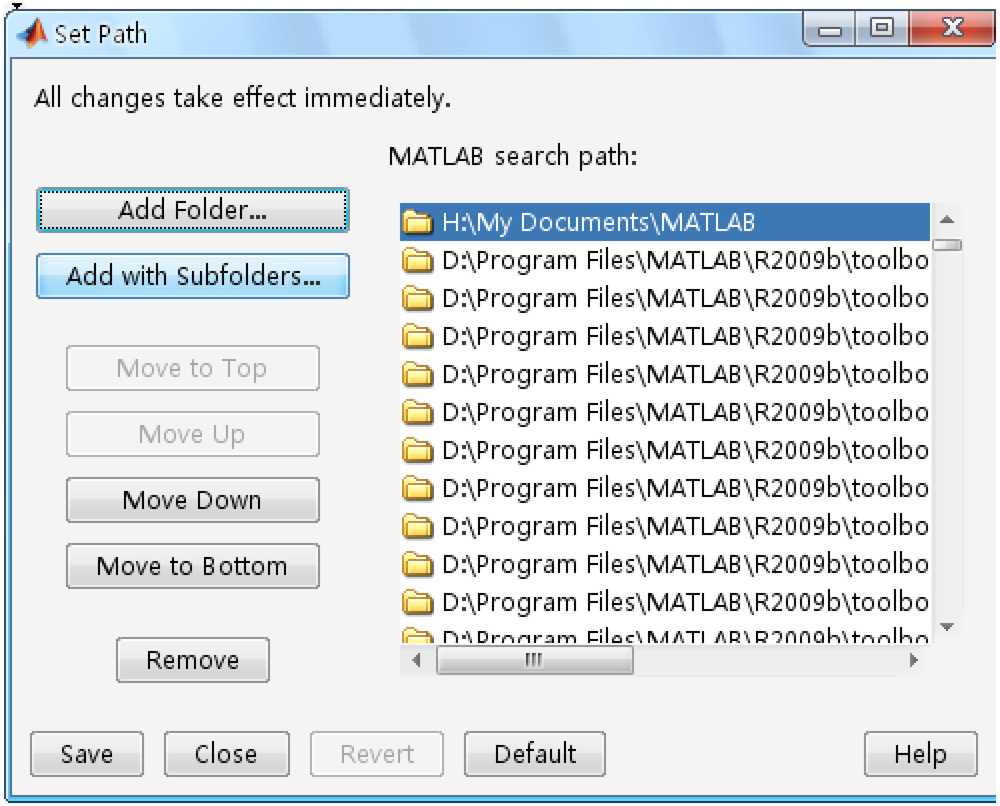
YALMIP工具箱的安装

- **Add the following directories to MATLAB path**
- /yalmip
- /yalmip/extras
- /yalmip/demos
- /yalmip/solvers
- /yalmip/modules
- /yalmip/modules/parametric
- /yalmip/modules/moment
- /yalmip/modules/global
- /yalmip/modules/sos
- /yalmip/operators

路径设置



MATLAB 7.9.0 (R2009b)
File Edit Debug Parallel Desktop Wind
New
Open... Ctrl+O
Close Command Window Ctrl+W
Import Data...
Save Workspace As... Ctrl+S
Set Path...
Preferences...
Page Setup...
Print... Ctrl+P
Print Selection...
1 H:\...AB\MCMCourse\opt5.m
2 H:\...ourse\YALMIPtest2.m
3 H:\...Course\YALMIPtest.m
4 H:\...B\MCMCourse\xxgh1.m
Exit MATLAB Ctrl+Q



Set Path
All changes take effect immediately.
MATLAB search path:
Add Folder...
Add with Subfolders...
Move to Top
Move Up
Move Down
Move to Bottom
Remove
H:\My Documents\MATLAB
D:\Program Files\MATLAB\R2009b\toolbo
Save
Close
Revert
Default
Help

安装测试与示例

- 测试YALMIP及已经存在的优化工具箱是否正常
- Yalmiptest
- 示例及入门
- Yalmipdemo（学习该工具箱的最佳途径）

实型变量—sdpvar

- sdpvar是YALMIP的一种核心对象，它所代表的是优化问题中的实型决策变量。
- 该种类型的变量可以通过如下方法产生：

➢ P=sdpvar(n,m); %表示x为n*m的矩阵

➢ P=sdpvar(n,n,’full’); %表示x为n*n的方阵

➢ P=sdpvar(n,n,'symmetric');%表示x为n*n的对称方阵

例：P=sdpvar(3,3);%方阵不带参数默认为对称方阵

P可以像正常的矩阵一样进行各种运算及操作，如：

v = trace(P)+P(1,1)+sum(sum(P))*5;

X = [P diag(P) v*eye(4)];

实型变量—sdpvar

- 变量sdpvar到底是什么？
- 它是符号型、未知的，但可以写为一组基矩阵的和
- P = P0+p1*P1+p2*P2+...+pn*Pn
- 可以用see函数来了解sdpvar类型数据的构成

其它决策变量类型

- intvar——整型

如：P=intvar(3,4)

- binvar——0-1型

如：P=binvar(5,5)

约束条件—set

- set（集合）是YALMIP的另外一种关键对象，用它来囊括优化问题的所有约束条件。
- 最常用的集合构造方法为采用set函数

例：

```
P = sdpvar(3,3);
F = set(P > 0);%唯一的不等式约束
```

对于存在上、下界的情况，也可以有如下的写法：

```
F = set(0 < diag(P) < 5);
```

约束条件—set

- 如果是等式约束，注意用“==”表示，如：

F = set(diag(P) == zeros(3,1));%等式约束

上式也可以写为F = set(diag(P) == 0);

- 如果问题包含多个约束条件，可以将多个条件用“+”相连

F = set('P>0') + set(0 < diag(P) < 5);

约束条件的三种构造方法

- P = sdpvar(3,3);
- F1 = set(P>0)+set(sum(P.^2) <[5,6,7]);
- F2 = set('P>0')+set('sum(P.^2) <[5,6,7]');
- F3 = [P>0, sum(P.^2) <[5,6,7]];
- 以上构造出的F1、F2、F3完全相同

求解函数—solvesdp

- 函数solvesdp用来求解优化问题
- s=solvesdp(F);%求解可行解问题
- s=solvesdp(F, f);%求解一般优化问题，其中f为目标函数
- s=solvesdp(F, f, options); %设定选项，比如选择算法等

结果获得

- 求解完成后，用P=double(P)提取解矩阵

示例 求解下列线性规划问题

$$\min(-2x_1 - x_2 - 4x_3 - 3x_4 - x_5)$$

$$s.t.\begin{cases} 2x_2 + x_3 + 4x_4 + 2x_5 \le 54 \\ 3x_1 + 4x_2 + 5x_3 - x_4 - x_5 \le 62 \\ x_1, x_2 \ge 0, x_3 \ge 3.32, x_4 \ge 0.678, x_5 \ge 2.57 \end{cases}$$

求解示例的源代码

```
f=-[2 1 4 3 1]'; A=[0 2 1 4 2; 3 4 5 -1 -1];
B=[54; 62];
xm=[0,0,3.32,0.678,2.57]';
P=sdpvar(5,1);
g=f'*P;
F=set(A*P<=B)+set(xm<=P);
sol=solvesdp(F,g);
P=double(P)
```

整数规划

- 将示例中的线性规划改为整数规划，则只需将程序中的变量定义部分稍做修改即可：

```
f=-[2 1 4 3 1]'; A=[0 2 1 4 2; 3 4 5 -1 -1];
B=[54; 62]; Ae=[]; Be=[];
xm=[0,0,3.32,0.678,2.57]';
P=intvar(5,1);
g=f'*P;
F=set(A*P<=B)+set(xm<=P);
sol=solvesdp(F,g);
P=double(P)
```

0-1规划

- 如果将示例的问题变为0-1规划问题，源程序应该如何改动？
- binvar()

混合规划问题

思考并实验：应该如何实现？